모바일 페이먼트를 위한 PUF 기반 OTP토큰

김 동 규*

1. 서 론

사용자 인증은 사용하고자 하는 시스템에 접근하기 위해 자신이 인가된 사용자라는 것을 증명하는 과정이다. 사용자 인증은 인증 방식에 따라 크게 인증서와 같이 사용자가 가지고 있는 것을 이용한 소유 기반 인증, 패스워드나 PIN처럼 사용자의 기억에 기반을 둔 지식 기반 인증, 홍채나 지문과 같은 사용자 고유의 생체 정보를 이용한 생체 기반 인증으로 나뉜다. 금융 결제처럼 높은 보안성을 필요로 하는 시스템에서는 두 가지의 인증 방식을 동시에 사용하는 이중 인증 방식(Two-factor authentication)으로 지식 기반 인증 기법인 패스워드와 보안 카드와 같은 소유 기반 인증 기법이 주로 결합되어 사용된다.

OTP(One-Time Password)[1]는 보안 카드의 문제점을 개선하여 이를 대체해 나가고 있는 대표적인 소유 기반 인증 기법이다. 보안 카드는 20~30가지의 한정된 색인(index)과 이에 해당하는 숫자값이 기록된 카드이므로 외부 공격자에게 패스워드와 같이 단 몇 번의 노출로 인해 그 정보가 드러나 이를 재사용하여 사용자로 위장할 수 있는 취약점이 있다. OTP는 매 세션 또는 정해진 시간마다 바뀌는 일회용의 패스워드를 사용하므로 정보가 노출되더라도 그 정보는 매우 짧은 시간 동안만 유효해 재사용 공격으로 인해 발생하는 사용자 피해를 차단할 수 있다.

그러나 OTP도 인증 서버에 대한 해킹과 OTP 토큰(제품)에 대한 침입 공격(Tampering attack)에 의해 생성 정보가 노출될 수 있다. OTP 토큰을 인증하기 위해 동일한 생성 정보로 OTP값을 생성하는 인증 서버를 해킹하거나, OTP 토큰을 해체하여 내부의 메모리 등을 공격함으로서 OTP 생성에 중요한 비밀 정보를 알아낸다면 공격자가 유효한 OTP 값을 재생성할 수 있다.

본고에서는 이러한 OTP의 취약점에 대해 분석하고, 이를 보완할 수 있는 하드웨어 OTP에 대해 설명한다. 그리고 기존의 보안수단(SE)인 USIM과 결합하여 큰 수정 없이 사용 가능한 모바일 페이먼트에서 활용할 수 있는 방안에 대해 논의하고자 한다.

2. OTP(One-Time Password)

OTP는 비밀번호 입력을 수행할 때마다 이전과 다른 새로운 비밀번호를 생성하여 인증하는 인증 수단이다. 이 장에서는 OTP 값의 생성 방식 및

※ 교신저자(Corresponding Author) : 김동규, 주소 : 서울특별시 성동구 왕십리로 222 융합전자공학부, 전화: 02-2220-2312, FAX: 02-2220-2312, E-mail : dqkim@hanyang.ac.kr

생성 알고리즘, 동기화 및 인증 방식, 기존 OTP에 대한 현황을 알아본다.

2.1 OTP 생성 방식 및 생성 알고리즘

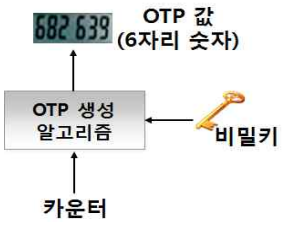

그림 1. OTP 생성 모듈

그림 1은 OTP 생성 모듈의 일반적인 형태이다. 인증 서버와 동기화되고 고정된 랜덤수인 비밀키와 세션마다 바뀌는 데이터 값인 카운터 값을 입력으로 OTP 생성 알고리즘을 통해 생성된 출력값을 6~8자리의 10진수 형태로 변환하여 출력한다.

표. 1 OTP 생성 알고리즘〔2〕

권고 알고리즘	암호키 길이(비트)	보안 강도 (비트)	비고
HMAC-SHA1	162	162	
HMAC-SHA256	256	256	
3DES	168	112	3key
AES	128/256	128/256	
SEED	128	128	
HIGHT	128	128	

표 1은 OTP 생성 알고리즘으로 권장되는 암호 알고리즘으로 HMAC-SHA1, 3DES, AES 등이 있다. 이외에도 112비트 이상의 보안 강도를 제공하는 암호 알고리즘을 사용할 수 있다.

2.2 동기화 및 인증 방식

인증 세션마다 바뀌는 카운터 값을 인증 서버와 동기화하는 방식은 S/Key 방식[3], 시간동기화 방식[4], challenge-response 방식[5], 이벤트 동기화 방식[6]으로 나뉜다.

2.2.1 S/Key 방식

S/Key 방식은 해쉬 함수를 이용하여 OTP 값을 생성하는 것으로 벨 통신 연구소에서 개발한 OTP 생성 방식이며 주로 유닉스 계열 운영체제에서의 인증을 위해 사용된다.

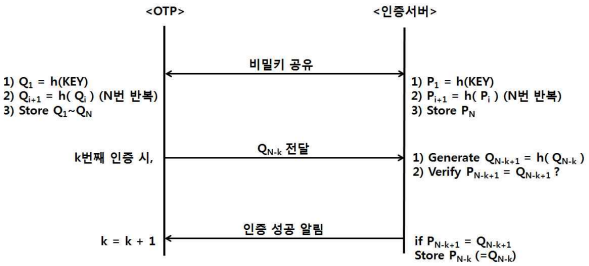

그림 2. S/Key 방식의 OTP 인증과정

그림 2는 S/Key 방식의 OTP 인증과정이다. 초기화시 클라이언트(OTP)는 인증 서버와 비밀키를 공유하고 공유된 비밀키를 해쉬 함수의 입력값으로 해쉬 값을 생성한다. 그리고 생성된 해쉬 값을 다시 해쉬 함수의 입력값으로 해쉬 값을 생성하는 과정을 n번 반복한다. 클라이언트는 생성된 n개의 해쉬 값을 저장하고 인증 서버는 마지막에 생성된 해쉬 값만을 저장한다. 첫 번째 OTP 인증시에, 클라이언트는 N-1번째로 생성된 해쉬 값을 인증 서버로 전송하고 인증 서버는 이 해쉬 값을 입력으로 한번 더 해쉬 함수를 실행한다. 이후 생

성된 해쉬 값을 인증 서버에 저장되어 있던 N번째 해쉬 값과 비교하고 해쉬 값이 동일하면 클라이언트에 인증 성공을 통보한다. 또한 다음 인증3에 사용하기 위해 클라이언트로부터 받은 N-1번째 해쉬 값을 저장하고 기존 N번째 해쉬 값은 삭제한다. 클라이언트는 인증 성공 수신 후 카운터(k)를 하나 증가 시켜 이후 인증에서는 N-k번째 해쉬 값을 사용한다.

S/Key OTP 알고리즘은 해쉬 값으로부터 해쉬 함수의 입력 값을 알아내기 어렵다는 특징을 이용한 알고리즘이다. 인증절차 중간에 클라이언트가 인증 서버로 전송하는 N-k번째 해쉬 값을 공격자가 알아내도 그 다음 인증에 사용되는 N-(k+1)번째 해쉬 값을 알아내는 것은 매우 어렵다. 그러나 클라이언트에 저장되어 있던 해쉬 값을 모두 인증에 사용하면 다시 초기화 과정이 필요하다.

2.2.2 시간 동기화 방식

시간 동기화 방식은 인증 서버와 OTP토큰이 동기화된 시간을 OTP 생성 알고리즘의 입력 값으로 사용한다.

그림 3. 시간 동기화 방식 OTP 인증 과정

그림 3은 시간 동기화 방식 OTP의 동기화 및 인증 과정이다. 우선 OTP와 인증 서버는 비밀키를 공유하고 시간을 동기화한다. 이후 OTP 인증을 시도할 때 OTP 토큰은 비밀키와 시간을 입력 값으로 OTP값을 생성하고 인증 서버로 전송한다. 인증 서버는 OTP 토큰과 동기화된 시간 값과 비밀키를 입력 값으로 동일 알고리즘을 수행하여 OTP값을 생성하고 OTP 토큰으로 부터 받은 OTP값과 비교하여 인증을 수행한다.

시간 동기화 방식은 시간을 OTP 생성 알고리즘의 입력 값으로 사용하므로 인증 서버와 OTP 토큰 사이의 별도의 데이터 전송이 필요 없다. 하지만 인증 서버와 OTP 토큰의 시간 동기화가 잘 맞지 않을 수 있어 1~2분정도의 오차를 허용한다.

2.2.3 Challenge-response 방식

Challenge-response 방식은 서버에서 임의의 난수를 OTP 토큰으로 보내고 이를 OTP 생성 알고리즘의 입력 값으로 사용해 OTP 값을 생성하는 방식이다.

그림 4. Challenge-response 방식 OTP의 인증과정

그림 4는 challenge-response 방식 OTP의 동기화 및 인증 과정이다. OTP값 생성 시 인증 서버에서 난수를 생성해 초기화시 OTP 토큰과 공유된 비밀키로 암호화 여 OTP 토큰으로 전송하고 OTP 토큰은 이 난수와 비밀키를 OTP 생성 알고리즘의 입력 값으로 사용하여 OTP값을 생성한다. 이후 OTP 토큰에서 생성된 OTP값은 인증 서버로 전송되고 인증 서버는 동일한 알고리즘과 입력 값을 사용하여 OTP값을 생성하고 이를 OTP로 부터 받은 OTP값과 비교하여 인증을 수

행한다.

Challenge-response 방식은 매번 다른 난수를 입력 값으로 사용하므로 challenge값을 얻어 낸다 해도 공격에 이용할 수 없어 보안성이 강하지만 OTP 토큰에서 challenge 수신을 위해 매번 인증 서버와의 네트워크 접속이 필요하다.

2.2.4 이벤트 동기화 방식

이벤트 동기화 방식은 OTP 토큰과 인증 서버와의 인증 횟수를 입력 값으로 사용하여 OTP 값를 생성한다.

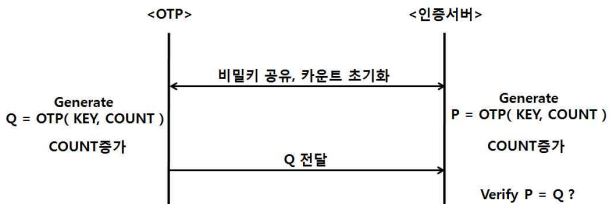

그림 5. 이벤트 동기화 방식 OTP 인증 과정

그림 5는 이벤트 동기화 방식 OTP의 인증 과정이다. OTP 토큰과 인증 서버는 초기화시 비밀키를 공유하고, 카운터를 초기화 한다. 이후 인증 수행시 OTP 토큰에서는 비밀키와 카운터값을 OTP 생성 알고리즘 입력값으로 사용하여 OTP 값을 생성하고 이를 인증 서버로 전달한다. 인증 서버에서는 OTP 토큰과 같은 알고리즘을 사용하여 비밀키와 카운터 값을 입력 값으로 OTP값을 생성하고 OTP로 부터 수신한 OTP값과 비교하여 인증을 수행한다. OTP 토큰과 인증 서버는 OTP값을 생성한 뒤 카운터 값을 증가시킨다.

이벤트 동기화 방식은 OTP 토큰이 OTP값을 생성하고 이를 인증에 사용하지 않으면 인증 서버와 OTP 토큰 간의 카운터 값이 일치하지 않는 문제가 발생할 수 있다. 이를 해결하기 위해서는 카운터 값에 대한 오차를 어느 정도 허용하는 방법과 카운터 값이 어긋났다고 판단될 경우 연속된 OTP값을 받아 유효성을 판단하는 방법이 있다.

2.3 기존 OTP 현황

국내 OTP 사용 현황과 제품 현황에 대해 알아본다.

2.3.1 국내 OTP 사용 현황

국내 인터넷 뱅킹은 공인인증서와 보안카드 체제가 주를 이루었으나 최근 계속되는 보안카드 해킹사고로 인해 OTP에 대한 사용이 늘어나고 있다.

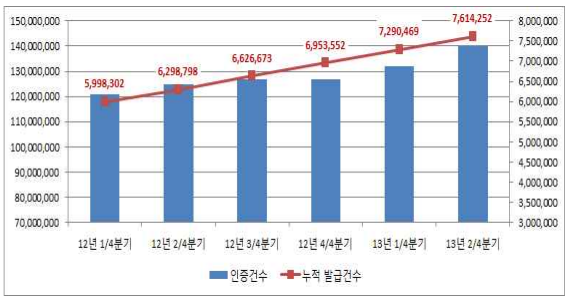

그림 6. 2013년 2/4분기 OTP 거래관련 통계 자료[7]

그림 6은 2013년 2/4분기의 OTP의 누적 발급건수와 OTP를 이용한 인증 건수이다. OTP를 사용한 인증건수와 OTP 누적 발급건수가 꾸준히 증가하고 있는 것을 볼 수 있다. 2013년 2/4분기에 이르러 OTP 인증건수는 1억4천만이 넘었으며, 누적 발급건수도 6백만이 넘었다. 인터넷 뱅킹이 보편화 되고, 해커에 의한 바이러스와 악성 코드의 감염에 의해 피싱, 파밍과 같은 인터넷뱅킹 피해가 증가함에 따라 이와 같은 OTP 사용 증가 추세는 꾸준히 지속될 전망이다.

2.3.2 OTP 제품 현황

OTP는 제품 형태에 따라 토큰형 OTP, 카드형 OTP, 그리고 모바일 OTP로 나뉜다.

(a) 토큰형 (b) 카드형 (c) 모바일

그림 7. 다양한 형태의 OTP

그림 7은 다양한 형태의 OTP를 나타낸다. 그림 7(a)는 일반적인 토큰형 OTP 제품의 모습으로 버튼과 LED가 달려있는 엄지손가락 크기의 플라스틱 덮개로 덮여있으며 내부에 MCU와 메모리가 칩 형태로 달려 있어 OTP 값을 생성한다. 토큰형 OTP는 현재 가장 많이 사용되는 OTP 형태이다. 그림 7(b)는 일반적인 신용카드 크기의 카드형 OTP제품의 형태로 토큰형 OTP와 기본적으로 같은 구성을 이루고 있지만, 신용카드 크기로 지갑 속에 휴대할 수 있어 휴대성이 강화된 타입이다. 그러나 공간의 제약에 따른 MCU와 배터리의 성능 제약으로 토큰형 OTP보다 OTP 생성 속도가 느리며 수명이 짧고 가격이 비싼 단점이 있다. 그림 7(c)는 스마트폰의 앱(App) 형태로 사용하는 모바일 OTP의 사용 형태이다. 모바일 OTP는 별도의 하드웨어 장비를 구입할 필요 없이 스마트폰에 앱을 설치하여 사용할 수 있으나 인증 서버 측에서 이 방식을 지원하지 않을 시 이용할 수 없으며 스마트폰과 운영체제에 따라 이용이 제한이 있을 수 있고 해킹에 취약한 단점이 있다.

국내외에서 일반적으로 사용되는 OTP 제품은 표 2와 같다. 국산 OTP의 데이터 동기화 방식은 주로 시간동기화 방식을 사용하며 대표적인 생산업체로는 미래테크놀로지, 이니텍사가 있다. 특히 이니텍사의 INISAFE Mobile OTP의 경우 시간과 이벤트 동기화 방식을 조합한 새로운 방식의 OTP를 도입하였다.

표 2. OTP 제품 현황

구분	회사명	제품명	생성방식	매체종류
국산	미래테크놀로지 [8]	MRT-400nP	시간동기화	OTP 전용기기
		MRT-500nP	시간동기화	OTP 전용기기
		MRC-100nP	시간동기화	카드형 OTP
	이니텍 [9]	INISAFE Mobile OTP	조합방식(T)	모바일 OTP
외산	RSA [10]	SecurID	시간동기화	OTP 전용기기 /모바일 /카드형
	SafeNet [11]	eToken PASS	시간동기화/ 이벤트동기화	OTP 전용기기
		eToken 3500	시간동기화	OTP 전용기기
		GOLD	Challenge-Response	OTP 전용기기
	VASCO [12]	Digipass	시간동기화/ 조합방식(T)	OTP 전용기기 / 카드형
	Active Identity [13]	Mini	시간동기화/ 이벤트동기화	OTP 전용기기

* 조합방식(T) : 시간동기화 중심의 조합방식

외산 OTP의 경우 데이터 동기화 방식으로 시간 동기화 방식 외에 이벤트 동기화 방식과 challenge-response 방식을 채용한 제품도 출시되고

있으며, 대표적인 업체로는 RSA, SafeNet, VASCO 등이 있다.

3. 기존 OTP 취약점

기존 OTP 시스템에서 OTP 생성 알고리즘의 주요 입력 데이터는 비밀키와 시간값 혹은 challenge와 같은 부가 정보로 이 값들은 OTP 토큰과 인증 서버 사이에 공유된다. 이 때 비밀키와 OTP 생성 알고리즘을 알면 공격자가 임의로 OTP 값을 재생산하는 것이 가능하다. 따라서 비밀키가 저장되는 곳인 서버와 OTP 기기 내 메모리가 주요 공격 목표가 된다.

3.1 서버 측 취약점

공격자가 인증 서버를 해킹하여 OTP 토큰의 생성 알고리즘과 비밀키를 습득하게 되면, 정상적인 OTP 사용자로 가장할 수 있다. 그림 8은 이러한 공격 시나리오를 나타낸다.

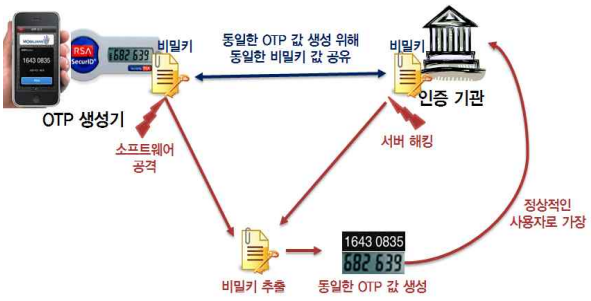

그림 8. 인증 서버 해킹으로 인한 취약점

실제로 2011년에 있었던 RSA사의 '시큐어 ID' 서버 해킹 사건[15] 으로 인해 RSA사의 '시큐어 ID' 제품의 키가 유출된 사례를 통해 서버에 저장되어 있는 정보의 안전성이 완전하지 않다는 것을 알 수 있다.

3.2 OTP 기기 측 취약점

OTP 토큰은 OTP값을 생성하기 위한 주요 정보를 메모리에 저장하고 이를 다른 내부의 IP와 같이 연결된 버스를 통해서 주고받는다. 메모리 소자나 버스 등은 물리적 공격에 의해 값을 알아내는 것이 가능하므로 주요 정보가 노출될 수 있다.

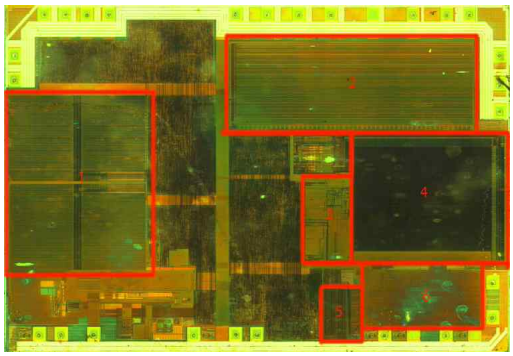

그림 9. 물리적 공격에 의한 레이아웃 분석

그림 9와 같이 물리적 공격을 통해 레이아웃을 분석하고 버스 프루빙, 메모리 공격 등의 공격을 수행할 수 있다. 실제로 NXP 칩이 물리적 공격에 의해 해킹[17]되어 해당 제품의 매출이 하락하는 사례도 존재하므로 OTP 기기 내 정보에 대한 안전성이 보완되어야 함을 알 수 있다.

4. Hardware OTP를 활용한 취약점 개선 방안

이번 장에서는 이전 장에서 분석했던 OTP의 취약점들을 해결하기 위한 해결방법으로 복제 불가능한 PUF(Physical Unclonable Func-tions)[16]를 비밀키로 이용하여 OTP값을 생성하는 HOTP(Hardware OTP)를 제안한다. 제안하는 HOTP는 OTP 생성 시 입력 값으로 사용되는 데이터를 인증 서버와 동기화 하는 방식에 따라 두

가지 방식으로 나누어진다.

4.1 시간 동기화 방식

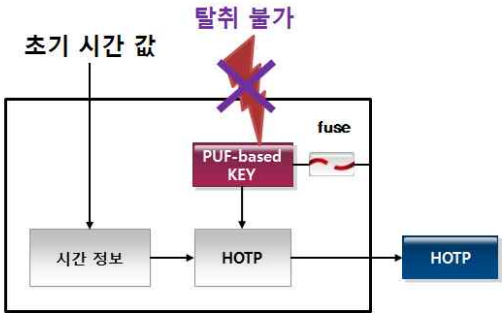

그림 10. 시간동기화 방식의 HOTP 내부 구조

그림 10은 PUF가 비밀키로 사용된 시간 동기화 방식의 OTP 토큰의 내부구조이다. 비밀키는 PUF로 구현되고 초기 인증 서버와 비밀키 공유 후에는 퓨즈가 끊겨 외부와의 접근이 차단된다. 다른 구성 요소는 기존 시간 동기화 방식 OTP와 동일하게 시간정보와 OTP 생성 모듈로 이루어진다.

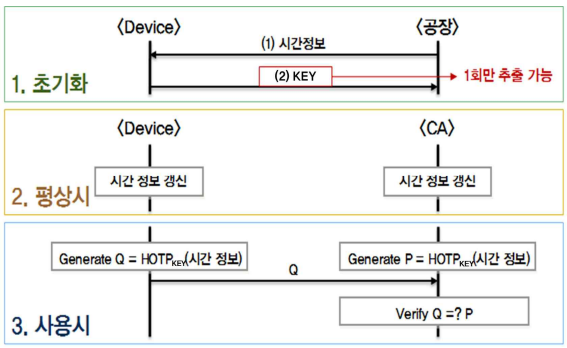

그림 11. 시간동기화 방식의 HOTP 동작과정

시간 동기화 방식 하드웨어 OTP의 동작과정은 그림 11과 같다. 공장 초기화시 HOTP에 시간 정보를 입력하고 HOTP로부터 비밀키 값을 추출한다. 이후 퓨즈를 끊어 향후 비밀키 값 추출을 방지 한다. 초기화 이후, 입력된 시간 초기값을 기준으로 내부의 Real-Time Clock(RTC)를 이용하여 현재의 시간 값을 갱신하며 대기한다. OTP 생성 요구 시에 시간 정보를 데이터로 입력하여 내부의 비밀키를 이용해 HOTP 값을 생성한다.

기존 OTP의 경우 OTP 내부 칩에 대한 탐지공격을 통해 비밀키를 알아낼 수 있는 취약점이 있었다. 그러나 HOTP의 경우 비밀키가 메모리에 저장되지 않고 PUF로 구현되어, OTP 생성 알고리즘 수행 시 칩 내부의 하드웨어 로직으로부터 직접적인 참조만 이루어지므로 비밀키를 외부에서 얻어 내는 것은 매우 어렵다. 또한 HOTP는 MCU가 존재하지 않고 모든 로직이 바이너리 코드가 아닌 순수 하드웨어로 구성되어 있어 공격자가 OTP 생성 알고리즘을 알기 위해서 로직을 분석하는 것은 불가능하다. 그러나 시간 동기화 방식의 HOTP는 인증 서버가 해킹 당했을 시에 인증 서버와 공유된 비밀키 값이 노출 될 수 있어 이를 이용해 공격자가 OTP 값을 재현 가능하다.

4.2 Challenge-response 방식

Challenge-response 방식의 HOTP 기존 challenge-response 방식의 OTP와 마찬가지로 challenge를 인증 서버로부터 수신하지만, 기존 대칭키 암호화 방식 대신에 공개키 암호화 방식을 이용하여 challenge를 암호화하여 보호한다. 인증 서버는 HOTP의 공개키로 challenge를 암호화하여 HOTP로 보내고 HOTP는 이를 받아 개인키로 복호화 한 후 OTP 생성 알고리즘의 입력 값으로 사용한다.

Challenge-response 방식의 HOTP의 내부구조는 그림 12와 같이 두개의 PUF 기반의 키와 공개키 생성을 위한 공개키 생성기, 공개키를 인증 서버에 암호화하여 전송하기 위한 대칭키 암호화 모듈, 인증 서버에서 공개키로 암호화하여 전

송한 challenge를 개인키로 복호화하기 위한 공개키 암호화 모듈, 그리고 challenge를 이용하여 OTP 값을 생성하기 위한 OTP 생성모듈로 구성된다.

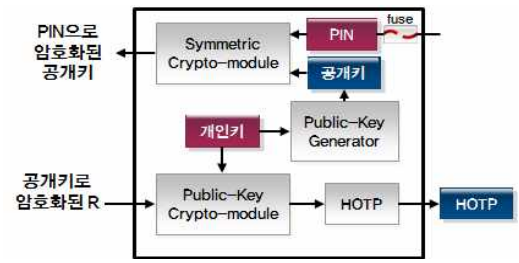

그림 12. challenge-response 방식의 HOTP 내부구조

HOTP 내에는 PUF 기반의 키가 두 개로 하나는 초기 인증 서버로 전송하는 공개키를 암호화하기 위한 위한 비밀키로 사용되는 PIN이며, 나머지 하나는 인증 서버로부터 암호화하여 수신된 challenge를 복호화하기 위한 개인키이다. PIN은 초기화 과정에서 한번만 추출된 이후 외부 인터페이스가 퓨즈로 차단되고, PUF로 구성되어 PIN을 공유한 인증 서버만이 암호화된 공개키를 복호화할 수 있다. 또한 PIN과 마찬가지로 PUF 기반의 개인키는 외부로 노출되지 않으므로 공격자는 개인키를 얻을 수 있는 방법이 없다.

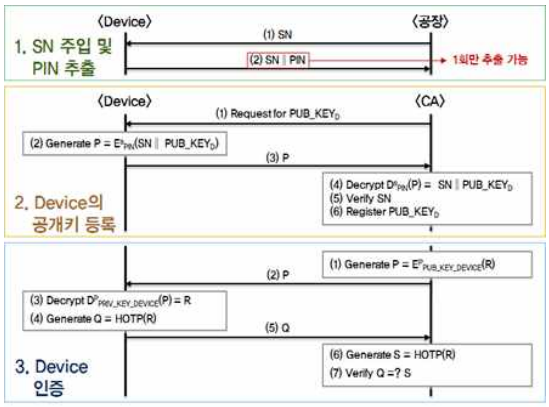

그림 13. challenge-response 방식의 하드웨어 OTP 동작과정

Challenge-response 방식의 HOTP 동작과정은 그림13과 같다. 초기화 시 HOTP에서 PIN 값이 추출되어 인증 서버(CA)에 전달되고 이후 퓨즈가 끊어져 더 이상의 PIN 추출이 방지된다. 인증 서버에 공개키 등록을 위해 HOTP 내부에서 공개키가 PIN을 비밀키로 사용하여 암호화 되고 온라인을 통해 인증기관서버로 전달된다. 인증 서버는 초기화시 하드웨어 OTP로 부터 전달된 PIN을 통해 공개키를 복호화 한다. OTP값 생성 시에는 인증 서버에서 랜덤 수 R을 공개키로 암호화하여 하드웨어 OTP로 전달하고 하드웨어 OTP는 이를 개인키로 복호화 한 후 OTP 생성 알고리즘의 입력값으로 사용한다. 인증 서버는 사용자가 입력한 OTP 값과 인증 서버에서 생성한 OTP 값을 비교하여 인증을 수행한다.

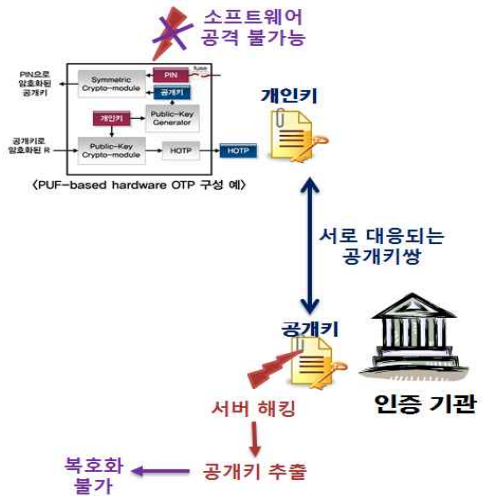

그림 14. 하드웨어 OTP의 보안 장점

기존의 OTP 시스템에서는 공격자가 OTP 인증 서버를 해킹하여 정상적인 사용자 OTP 토큰의 비밀키를 탈취하여 정상적인 사용자로 위장할 수 있으나, 그림 14와 같이 제안하는 challenge-response 방식의 하드웨어 OTP에서는 공개키 암호 시스템을 사용하므로 인증기관에서 서

버 해킹으로 인해 사용자의 공개키가 누출되어도 공격자는 인증 서버에서 공개키로 암호화하여 전송한 challenge를 복호화 할 수 없어 정상적인 사용자로의 위장이 불가능하다. 또한 서버 해킹에 의해 PIN이 노출되더라도 PIN은 기기의 공개키를 등록할 때에만 1회성으로 사용되므로 공격이 유효한 시점(사용자가 OTP를 사용 중)에는 노출된 PIN를 유효하게 사용하는 것이 불가능하다. 따라서 challenge-response 방식을 이용할 경우 앞선 시간동기화 방식의 취약점이었던 서버해킹의 위험성을 해결할 수 있을 것으로 기대된다.

5. HOTP의 응용

이번 장에서는 HOTP를 기존의 모바일 기기에 적용하여 모바일 페이먼트에서 활용하는 방법을 알아본다.

5.1 Two-chip Package Solution

HOTP는 작은 칩 형태로 제공되어 사용자가 토큰 형식으로 휴대할 필요 없이 기존 모바일기기 내부의 Secure Element(SE)와 물리적으로 연결하여 사용할 수 있다.

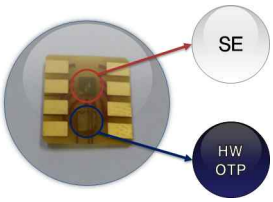

그림 15. Two-chip Package 구조

그림 15는 실제 HOTP와 SE에 해당하는 칩이 하나의 USIM 패키지에 결합된 모습이다. 이때 하드웨어 OTP와 기존 SE 내부의 칩이 다이 연결을 통해 하나의 패키지로 작동하며 SE 외부의 형태는 변형 되지 않아 기존 모바일 기기 형태를 수정하지 않으므로 시스템 구축비용이 저렴하다. 또한 공인인증서를 포함하여 각종 키가 저장되는 SE와 HOTP가 함께 인증 요소로 동작함으로써 이중인증이 가능하다.

5.2 모바일 페이먼트에서의 활용

5.2.1 모바일 뱅킹

모바일 뱅킹은 사용자의 모바일을 통해 계좌조회 및 계좌이체와 같은 금융 서비스를 받을 수 있는 결제 시스템이다. 기존의 모바일 뱅킹 이용 시에는 USIM 내의 공인인증서와 사용자가 가지고 있는 보안카드 혹은 OTP를 통해 결제를 진행한다. 이 때 공격자는 악성코드를 통해 사용자의 공인인증서를 탈취 할 수 있다. 또한 보안카드 번호 혹은 OTP 값을 모바일로 입력 할 때 키 로그 혹은 스크린 해킹을 통해 그 값을 중간에 얻어낼 수 있다. 이후 정상적인 사용자로부터 얻어낸 이 두 가지 인증요소를 통해 정상적인 사용자로 가장하여 금융 범죄를 저지를 수 있다.

그림 16 모바일 뱅킹에서의 HOTP 적용

그림 16과 같이 USIM 내부에 존재하는 USIM 칩과 HOTP를 연결하여 하나의 패키지로 동작 시킨다면 HOTP에서 생성된 OTP값이 사용자 입력 없이 자동으로 USIM으로 전달되어 기존 OTP값 입력 시에 발생 할 수 있는 키 로그 해킹 문제가 해결되며 사용자 편의성 또한 향상 된다. HOTP는 PUF 기반의 키를 사용하므로 공격자는 이를 복제 및 추출을 할 수 없다. 공격자가 USIM 내부의 공인인증서를 복제해도 PUF가 적용된 HOTP의 OTP값을 알 수 없으므로 정상적인 사용자로 위장할 수 없다.

5.2.2 전자지갑

전자 지갑은 스마트폰을 통해 모바일 신용카드 이용 시, 여러 금융사의 모바일 신용카드를 하나의 앱에서 관리하고 더불어 멤버십, 쿠폰 등도 함께 관리해 주는 통합 앱이다. 현재 전자 지갑은 사용자 인증을 위해 USIM칩 내 가입자 정보를 이용한다. 전자 지갑 앱 사용자와 스마트폰의 가입자가 다른 경우 사용자는 전자 지갑 앱을 사용할 수 없다. 그러나 USIM칩 내 가입자 정보를 해킹을 통해 수정하는 것이 가능하여 공격자가 자신의 스마트폰 USIM 칩 내 가입자 정보를 타인의 정보로 변경하고 타인의 신분으로 위장하여 전자지갑 앱을 통해 모바일 신용카드 및 각종 멤버십, 쿠폰 등을 사용할 수 있다.

그림 17. 전자지갑에서의 HOTP 적용

이러한 취약점을 개선하기 위해 전자지갑 시스템의 인증요소로 HOTP를 사용할 수 있다. 그림 17은 전자지갑에서 인증요소로 HOTP가 사용되는 것을 보여준다. 하드웨어 OTP를 사용할 경우 사용자 인증은 패키징 되어 있는 SE의 내부 정보뿐만 아니라 HOTP를 함께 사용하여 이루어지며, 인증기관은 사용자로 부터 SE의 정보와 정상적인 HOTP값을 동시에 수신하였을 시만 전자지갑 사용을 허가한다. 공격자는 USIM칩 내 가입자 정보를 수정하여도 PUF가 적용된 HOTP의 OTP값을 재현하지 못하므로 위장 인증이 불가능하다.

5.2.3 소액결제

소액결제는 온라인 결제 시 소액의 한도 안에서 간단한 인증절차를 걸쳐 빠르고 쉽게 결제를 진행할 수 있도록 해주는 결제 시스템이다. 기존의 소액결제에서 결제를 위해서는 핸드폰 번호, 주민 등록 번호, 통신사 종류 등의 정보가 필요하며 사용자 인증을 위해서 사용자 스마트폰으로 전달되는 SMS 인증 번호를 입력해야 한다. 하지만 이러한 인증 시스템에서는 공격자가 해킹 툴을 이용해 중간에 인증 번호를 가로챌 수 있으며 정상적인 사용자로 위장해 타인의 전화번호로 결제를 할 수 있다.

그림 18. 소액결제에서의 HOTP 적용

그림 18과 같이 기존 소액결제에 HOTP를 인증 수단으로 추가하면, 공격자가 결제를 위한 사용자의 정보와 인증 메시지를 가로채더라도 PUF가 적용된 HOTP의 OTP 값을 알 수 없으므로 타인의 전화번호로 결제를 할 수 없다. 또한, HOTP의 OTP값 전송은 백그라운드에서 사용자 인식 없이 수행 되므로 사용자 편의성은 기존과 동일하다.

5.2.4 모바일 신용카드

모바일 신용카드는 스마트폰의 USIM안에 신용카드를 발급받아 플라스틱 신용카드를 대신하여 스마트폰을 통해 온라인 및 오프라인 결제를 진행하는 시스템이다. 모바일 신용카드로 결제하기 위해서는 USIM 칩 내부의 카드 정보인 카드 번호, 유효기간, CVC등의 카드 정보와 사용자가 입력하는 카드 비밀번호가 필요하다. 하지만 카드 정보는 공격자가 악성코드를 통해 탈취할 수 있으며 사용자가 입력하는 카드 비밀번호도 키 로그를 통해 탈취가 가능하다. 공격자는 탈취한 카드정보와 카드 비밀번호를 통해 정상적인 사용자로 가장하여 범죄를 저지를 수 있다.

그림 19. 모바일 신용카드에서의 HOTP 적용

이러한 취약점을 해결하기 위해, 그림 19와 같이 USIM칩 내부에 HOTP를 삽입하여 모바일 신용카드 결제 시 추가적인 인증수단으로 하드웨어 OTP를 사용하면, 공격자가 정상적인 사용자의 카드정보와 카드 비밀번호를 획득하더라도 PUF가 적용된 하드웨어 OTP와 동일한 OTP값을 생성할 수 없으므로 인증에 성공할 수 없다.

6. 결론

OTP는 인증 시마다 다른 비밀번호를 사용하는 형태로, 인터넷 뱅킹이나 스마트폰 뱅킹에서 기존의 보안 수단인 보안 카드와 공인인증서의 취약점으로 인해 발생하는 해킹 등의 피해를 막을 수 있어 최근에 그 사용량이 증가하고 있다. 그러나 OTP 역시 제품 내부의 메모리에 비밀키와 같은 주요 인증 데이터를 저장하므로, 침입 공격을 통해 이러한 주요 인증 정보가 유출될 수 있다. 또한, 인증 서버의 해킹을 통해서도 해당 정보가 유출될 수 있다. 본고에서는 이러한 취약점을 분석하여 이를 보완할 수 있는 복제 불가능한 PUF를 이용한 HOTP를 제안하였다. 비밀키 등의 주요 인증 정보로 활용되는 PUF는 복제가 불가능하고, OTP 생성 모듈과 함께 하드웨어 OTP 내부의 하드웨어 로직 셀들로 구현되기 때문에, 메모리 스캔이나 버스 프루빙 등을 통해 분석이 어렵다. 동작 방식이 간단하고 기존의 OTP에 사용되는 방식인 시간 동기화 방식과, 공개키 암호화 방식을 사용하여 서버 해킹을 통한 피해를 방지할 수 있는 challenge-response 방식을 구현하였다.

HOTP는 소형의 칩 형태로 구현되어 기존의 보안 수단(인증서, 패스워드, PIN)이 구현된 칩과 two-chip package 형태로 결합되어 기존의 시스템에 최소한의 수정으로 보안 강화 수단으로 사용

될 수 있다. 특히, 모바일폰의 USIM과 two-chip package로 결합되어 모바일 뱅킹이나 온라인 신용카드, 소액 결제, 전자 지갑 등의 모바일 페이먼트 응용 수단에 활용할 수 있다.

참 고 문 헌

[1] N.Haller, C.Metz, "A One-Time Password System," Network Working Group RFC 1938, 1996.

[2] 정보통신단체표준, "일회용 패스워드(OTP) 알고리즘 프로파일", TTAK.KO-12.0193, 2012.

[3] N.Haller, "The S/KEY One-Time Password System," Network Working Group RFC 1760, 1995.

[4] D.M'Raihi, S.Machani, M.Pei, J.Rydell, "Time-Based One-Time Password Algorithm," Internet Engineering Task Force RFC 6238, 2011.

[5] D.M'Raihi, J.Rydell, S.Bajaj, S.Machani, D.Naccache, OCRA: OATH Challenge- Response Algorithm, Internet Engineering Task Force RFC 6287, 2011.

[6] D.M'Raihi, M.Bellare, F.Hoornaert, D.Nacc- ache, O.Ranen, HOTP: An HMAC-Based One-Time Password Algorithm," Network Working Group RFC 4226, 2005.

[7] 금융보안연구원 OTP통합인증센터, "2013년 2/4분기 OTP거래관련 통계자료", 2013.

[8] 미래테크놀로지-H/W OTP 개요, http://www.mirae-tech.co.kr/solution/solutions1_tab01.php

[9] 이니텍-INISAFE MOBILE OTP, http://www.initech.com/www/html/inisafe/goMenu3_5_1.html

[10] RSA-SecureID, http://www.emc.com/security/rsa-securid/rsa-securid-hardware-authenticators.htm

[11] SafeNet-eToken, http://www.safenet-inc.com/data-protection/authentication/otp-authentication

[12] VASCO-Digipass, http://www.vasco.com/ products/products.aspx

[13] ActiveIdentity-Mini, http://www.sentechgroup. com/node/95

[14] R. Anderson, M. Kuhn, "Tamper Resistance - a Cautionary Note", Proceedings of the Second Usenix Workshop on Electronic Commerce, pp. 1-11, 1996

[15] 김희연, "OTP 뚫렸다...RSA 시큐어 ID 사이버 공격당해" <http://www.zdnet.co.kr/news/ news_view.asp?artice_id=20110318173025>, 지디넷코리아, 2012

[16] R. Pappu, B. Recht, J. Taylor, and N. Gershenfeld, "Physical one-way functions," Science, Vol. 297 No.5589 pp.2026 - 2030, 2002

[17] https://www.blackhat.com/docs/sp-14/materials/arsenal/sp-14-Almeida-Hacking-MIFARE-Classic-Cards-Slides.pdf

김 동 규

- 1992년 2월 서울대학교 컴퓨터 공학과 학사
- 1994년 2월 서울대학교 컴퓨터 공학과 석사
- 1999년 2월 서울대학교 컴퓨터 공학과 박사
- 1999년 9월 ~ 2006 년 2월 부산대학교 조교수
- 2006년 3월 ~ 현재 년 2월 한양대학교 교수
- 2007년 1월 ~ 현재 한국 정보보호학회 상임이사
- 2007년 1월 ~ 현재 한국 정보처리학회 상임이사
- 2009년 1월 ~ 현재 한국 정보과학회 논문편집위원
- 2011년 1월 ~ 현재 한국 멀티미디어학회 재무부회장
- 관심분야 : 암호알고리즘, 보안 SoC 설계, 기능안전 및 보안